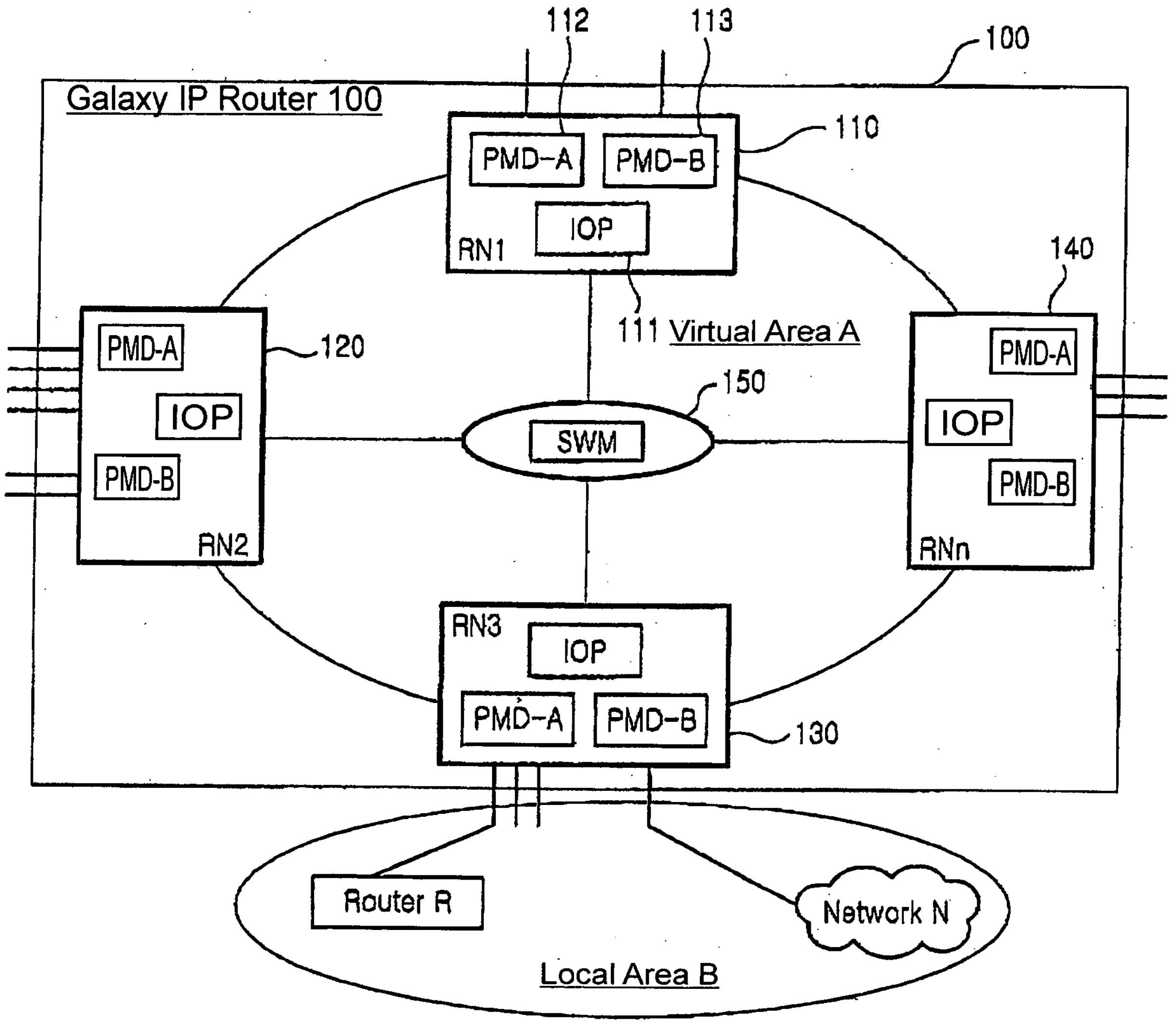

112
113
100
Galaxy IP Router 100
PMD-A
PMD-B
110
IOP
RN1
140
111 Virtual Area A
PMD-A
IOP
PMD-B
120
150
SWM
PMD-A
IOP
PMD-B
RN2
RNn
RN3
IOP
PMD-A
PMD-B
130
Router R
Network N
Local Area B

FIG. 1

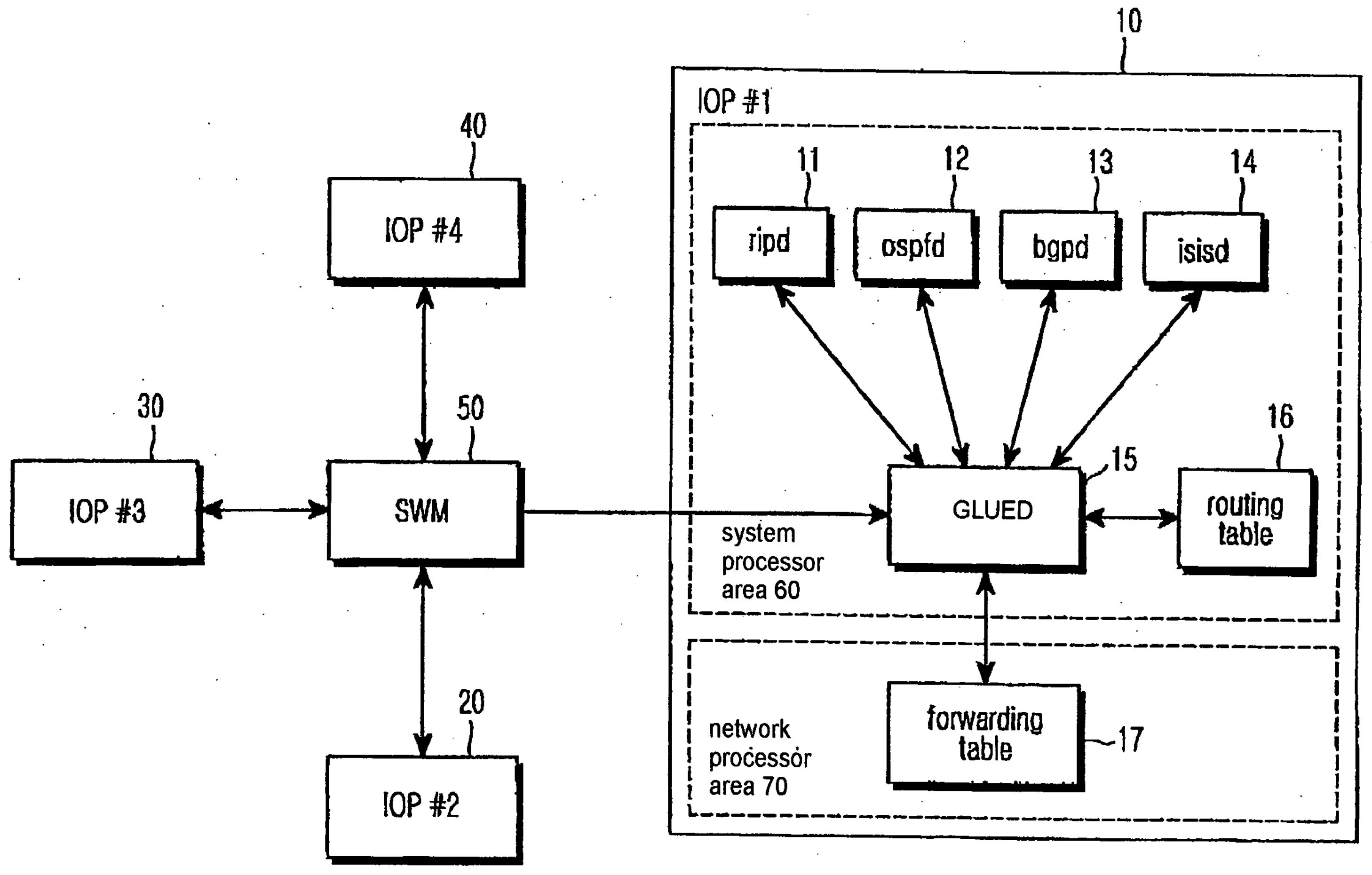

10
IOP #1
11
12
13
14
ripd
ospfd
bgpd
isisd
16
15
GLUED
routing table
system processor area 60
forwarding table
17
network processor area 70
40
IOP #4
30
50
IOP #3
SWM
20
IOP #2

FIG.2

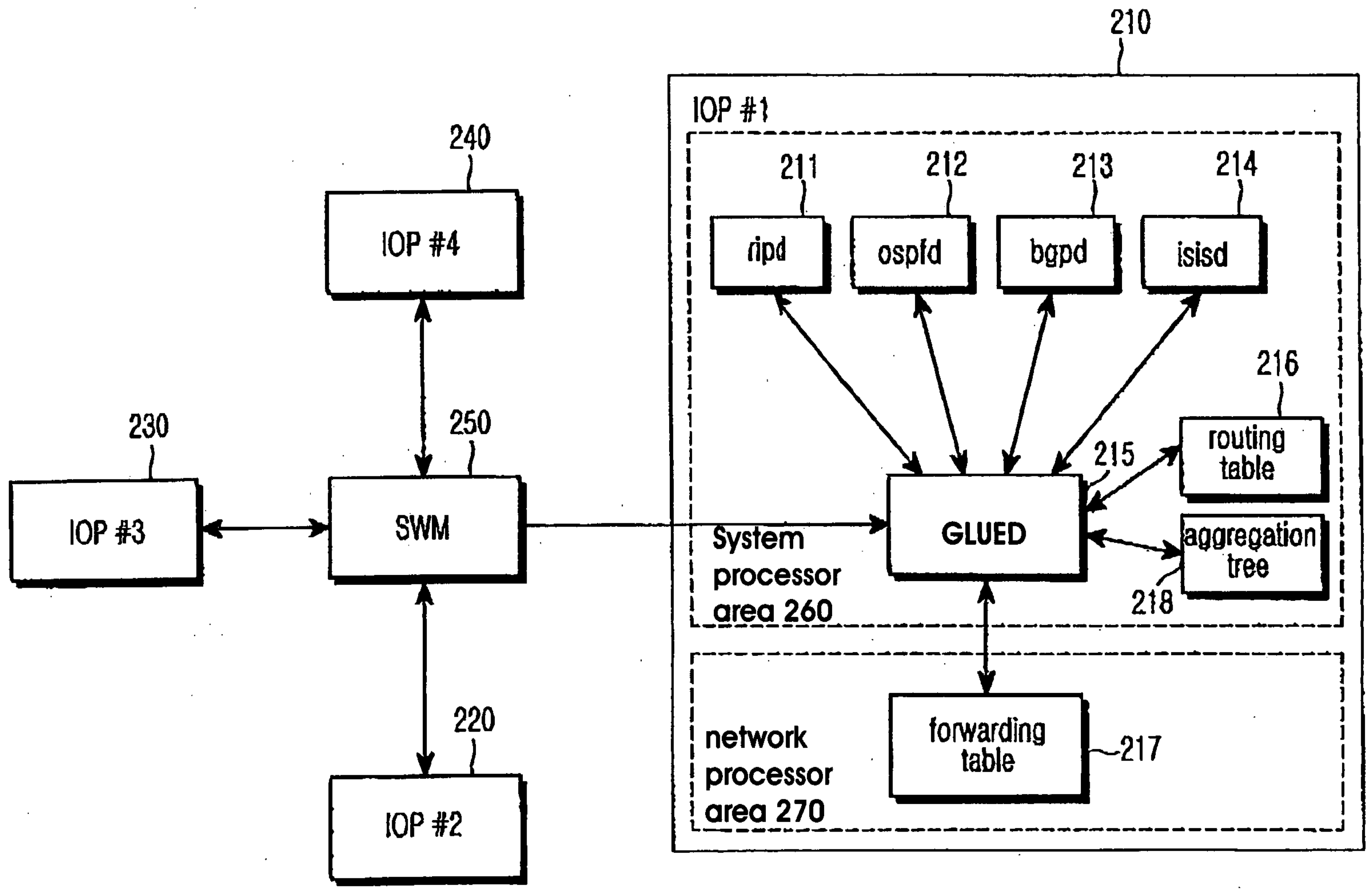

210
IOP #1
211
ripd
212
ospfd
213
bgpd
214
isisd
216
routing table
215
GLUED
aggregation tree
218
System processor area 260
forwarding table
217
network processor area 270
240
IOP #4
250
SWM
230
IOP #3
220
IOP #2

FIG.3

Prefix	Length	Type	Source IOP	IOP Flag	FT Flag

FIG.4A

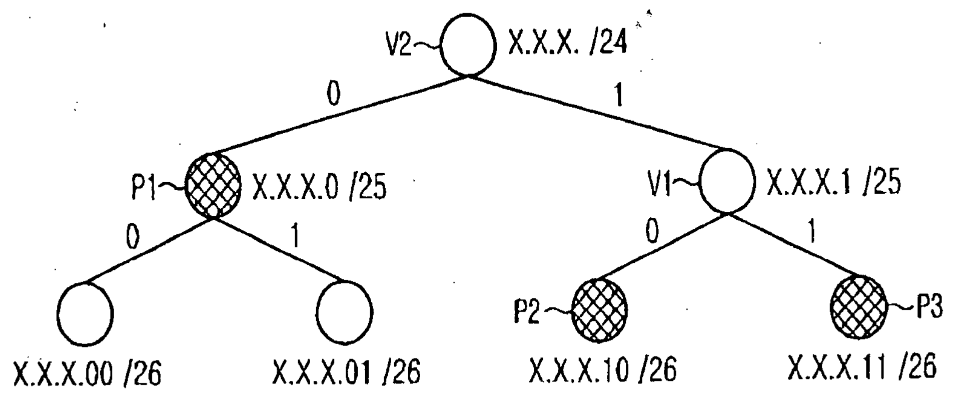
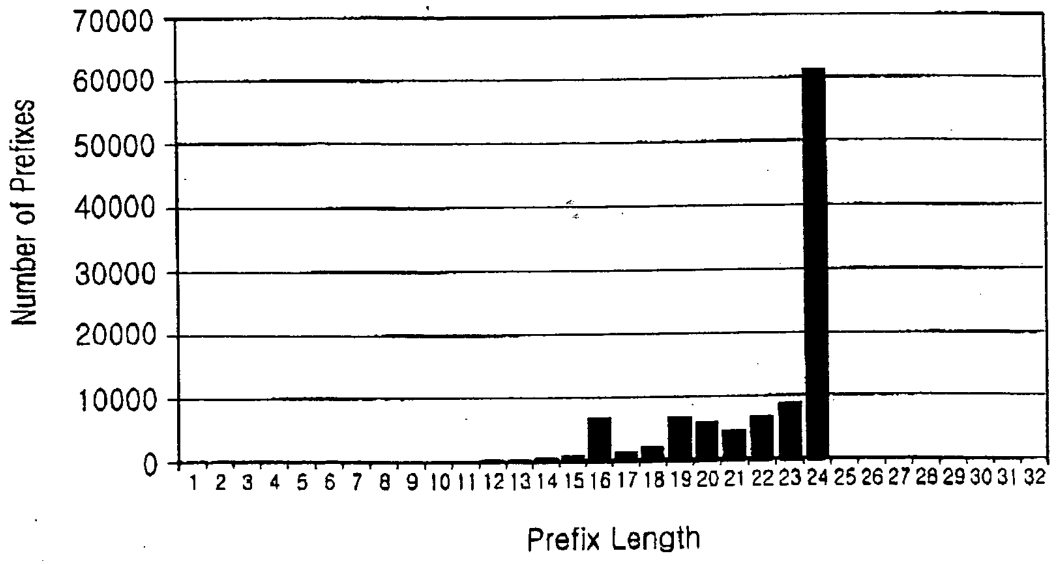


FIG.4B



70000
60000
50000
40000
30000
20000
10000
0
Number of Prefixes
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
Prefix Length

FIG.4C

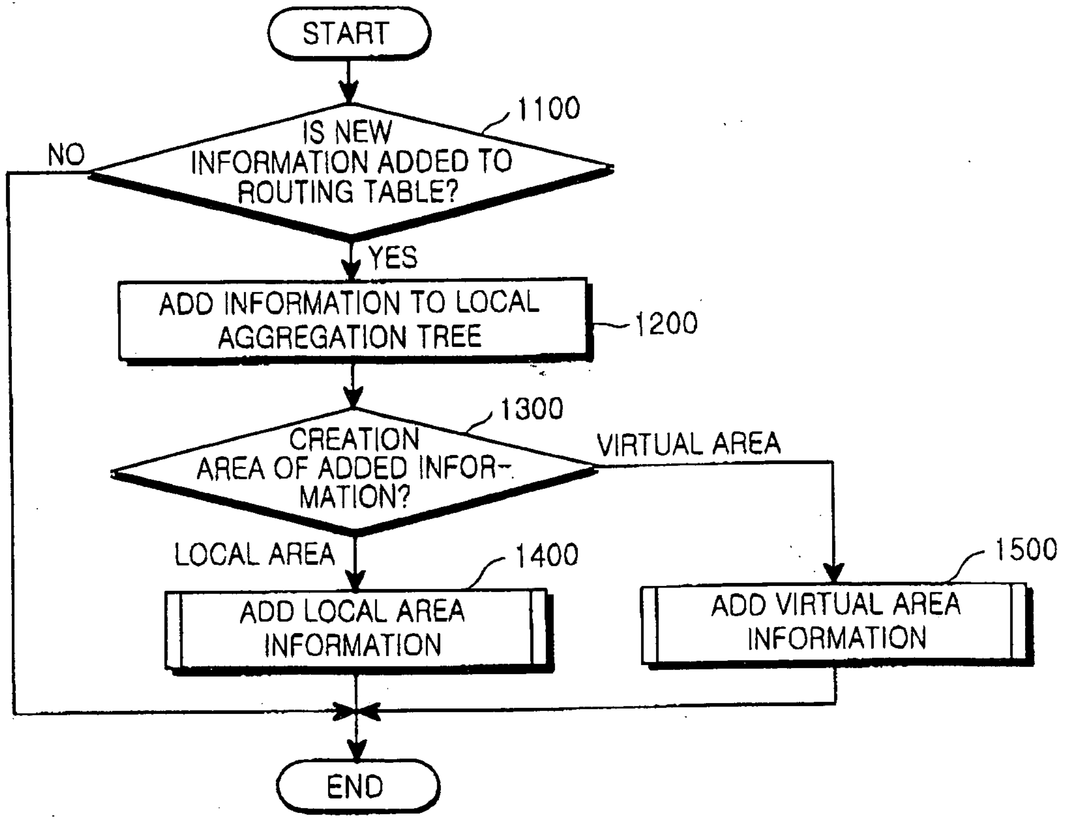

START
IS NEW INFORMATION ADDED TO ROUTING TABLE?
1100
NO
YES
ADD INFORMATION TO LOCAL AGGREGATION TREE
1200
CREATION AREA OF ADDED INFORMATION?
1300
VIRTUAL AREA
LOCAL AREA
ADD LOCAL AREA INFORMATION
1400
ADD VIRTUAL AREA INFORMATION
1500
END

FIG.5

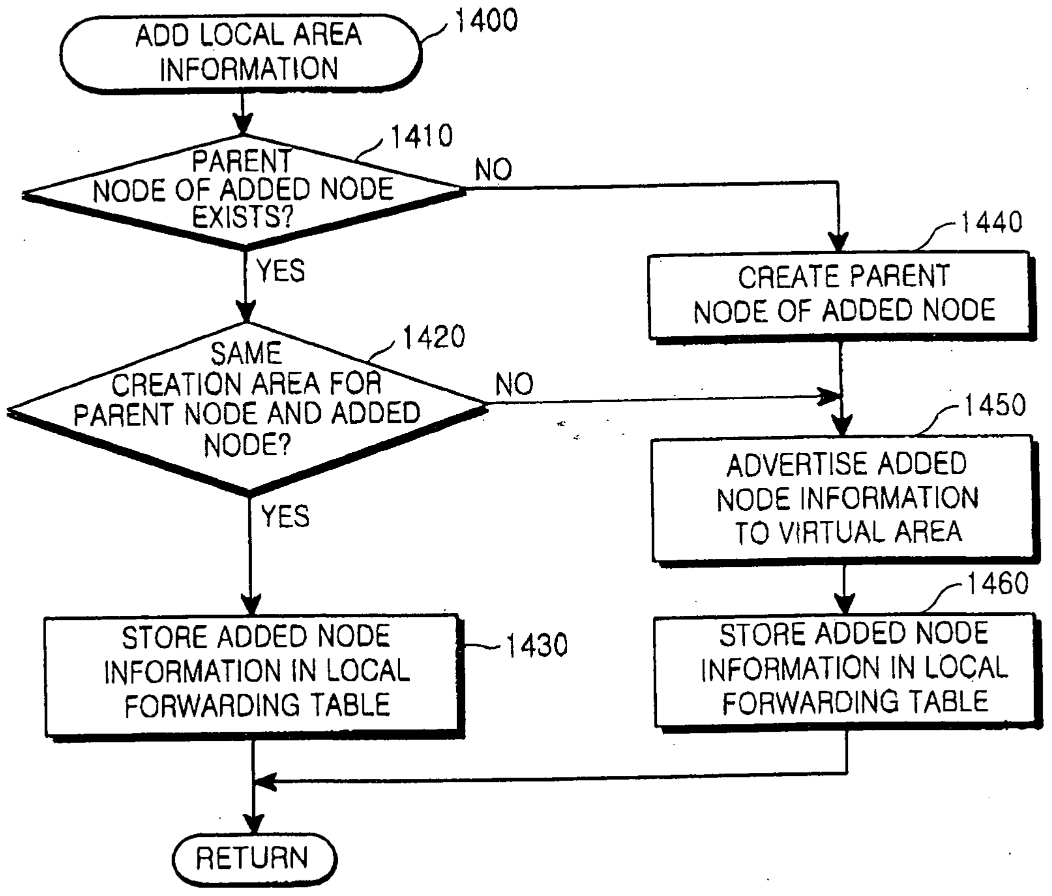

ADD LOCAL AREA INFORMATION
1400
PARENT NODE OF ADDED NODE EXISTS?
1410
NO
YES
CREATE PARENT NODE OF ADDED NODE
1440
SAME CREATION AREA FOR PARENT NODE AND ADDED NODE?
1420
NO
YES
ADVERTISE ADDED NODE INFORMATION TO VIRTUAL AREA
1450
STORE ADDED NODE INFORMATION IN LOCAL FORWARDING TABLE
1430
STORE ADDED NODE INFORMATION IN LOCAL FORWARDING TABLE
1460
RETURN

FIG.6

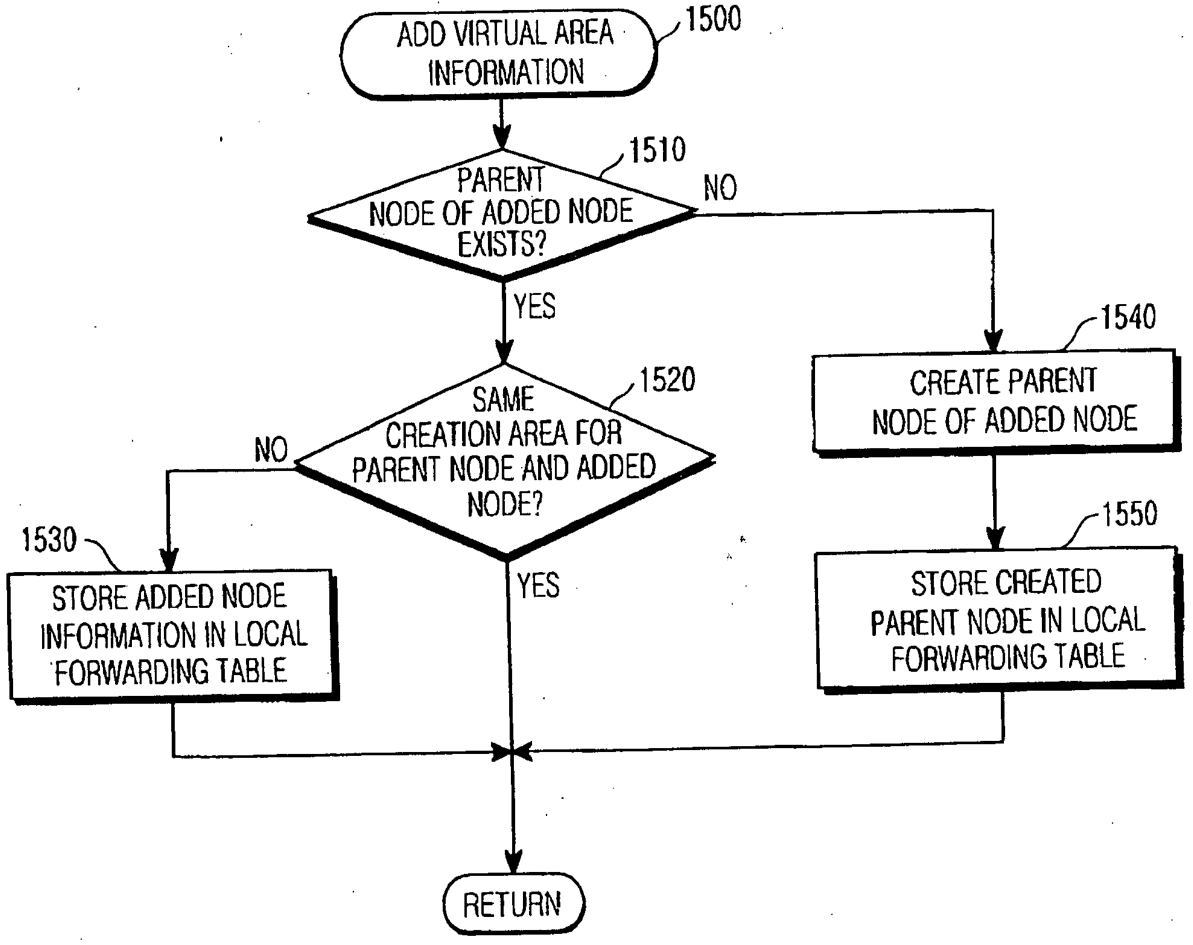

ADD VIRTUAL AREA INFORMATION
1500
PARENT NODE OF ADDED NODE EXISTS?
1510
NO
YES
SAME CREATION AREA FOR PARENT NODE AND ADDED NODE?
1520
NO
YES
STORE ADDED NODE INFORMATION IN LOCAL FORWARDING TABLE
1530
CREATE PARENT NODE OF ADDED NODE
1540
STORE CREATED PARENT NODE IN LOCAL FORWARDING TABLE
1550
RETURN

FIG.7

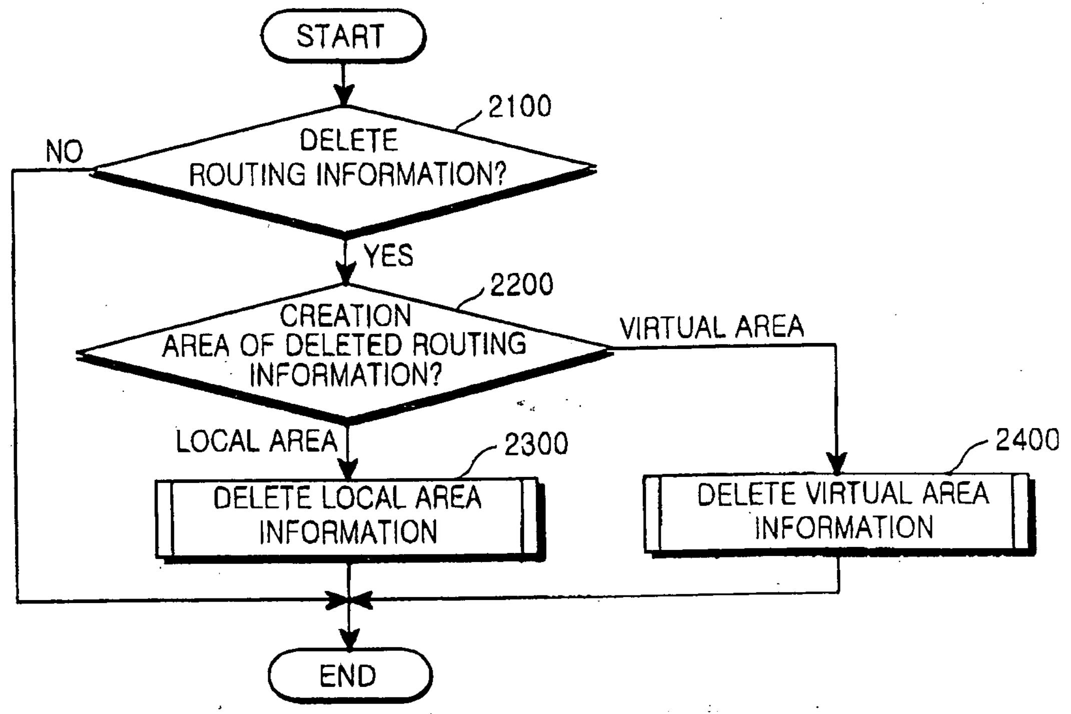

START
2100
DELETE ROUTING INFORMATION?
NO
YES
2200
CREATION AREA OF DELETED ROUTING INFORMATION?
VIRTUAL AREA
LOCAL AREA
2300
DELETE LOCAL AREA INFORMATION
2400
DELETE VIRTUAL AREA INFORMATION
END

FIG.8

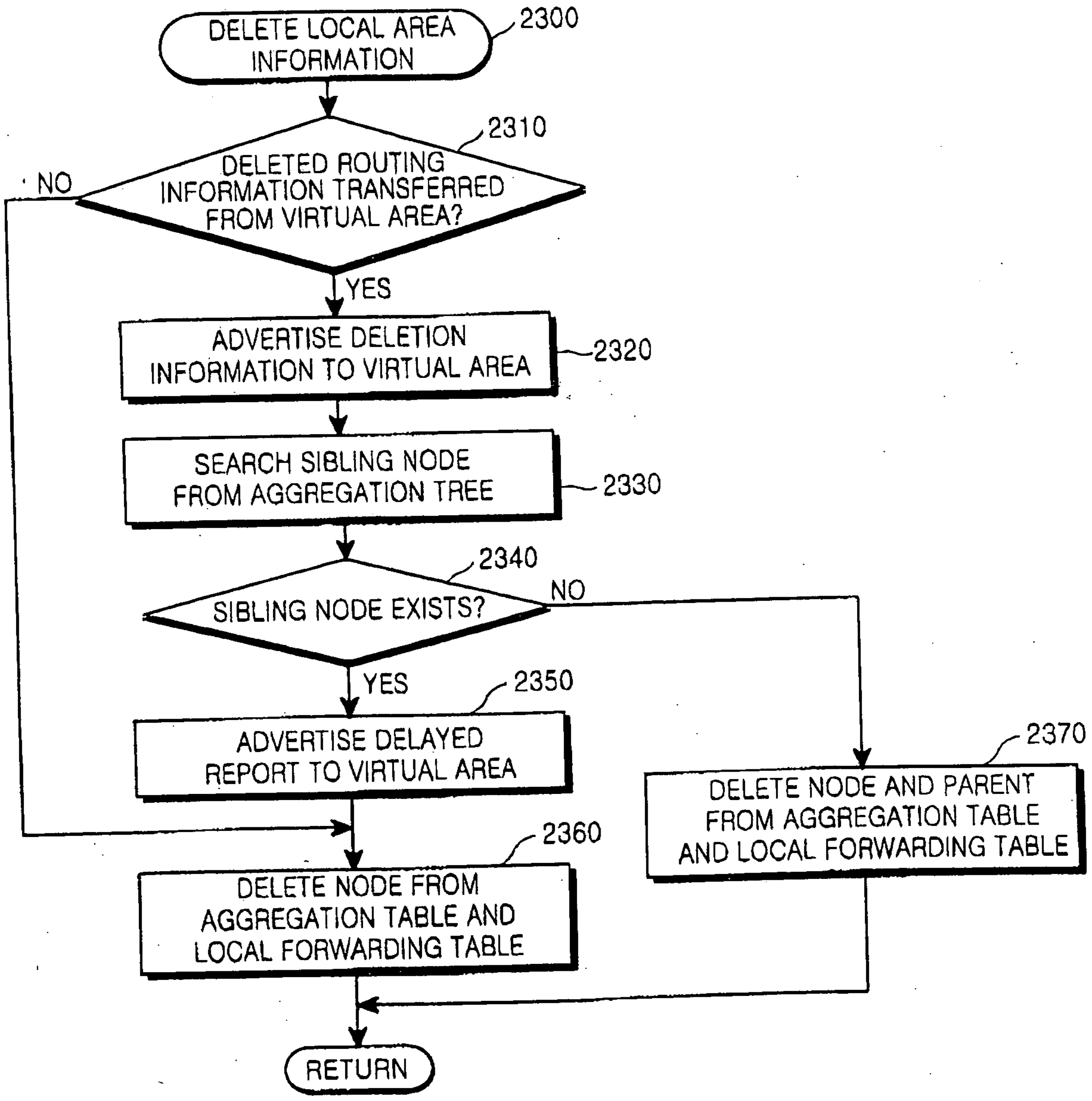

DELETE LOCAL AREA INFORMATION
2300
2310
DELETED ROUTING INFORMATION TRANSFERRED FROM VIRTUAL AREA?
NO
YES
ADVERTISE DELETION INFORMATION TO VIRTUAL AREA
2320
SEARCH SIBLING NODE FROM AGGREGATION TREE
2330
2340
SIBLING NODE EXISTS?
NO
YES
2350
ADVERTISE DELAYED REPORT TO VIRTUAL AREA
2370
DELETE NODE AND PARENT FROM AGGREGATION TABLE AND LOCAL FORWARDING TABLE
2360
DELETE NODE FROM AGGREGATION TABLE AND LOCAL FORWARDING TABLE
RETURN

FIG.9

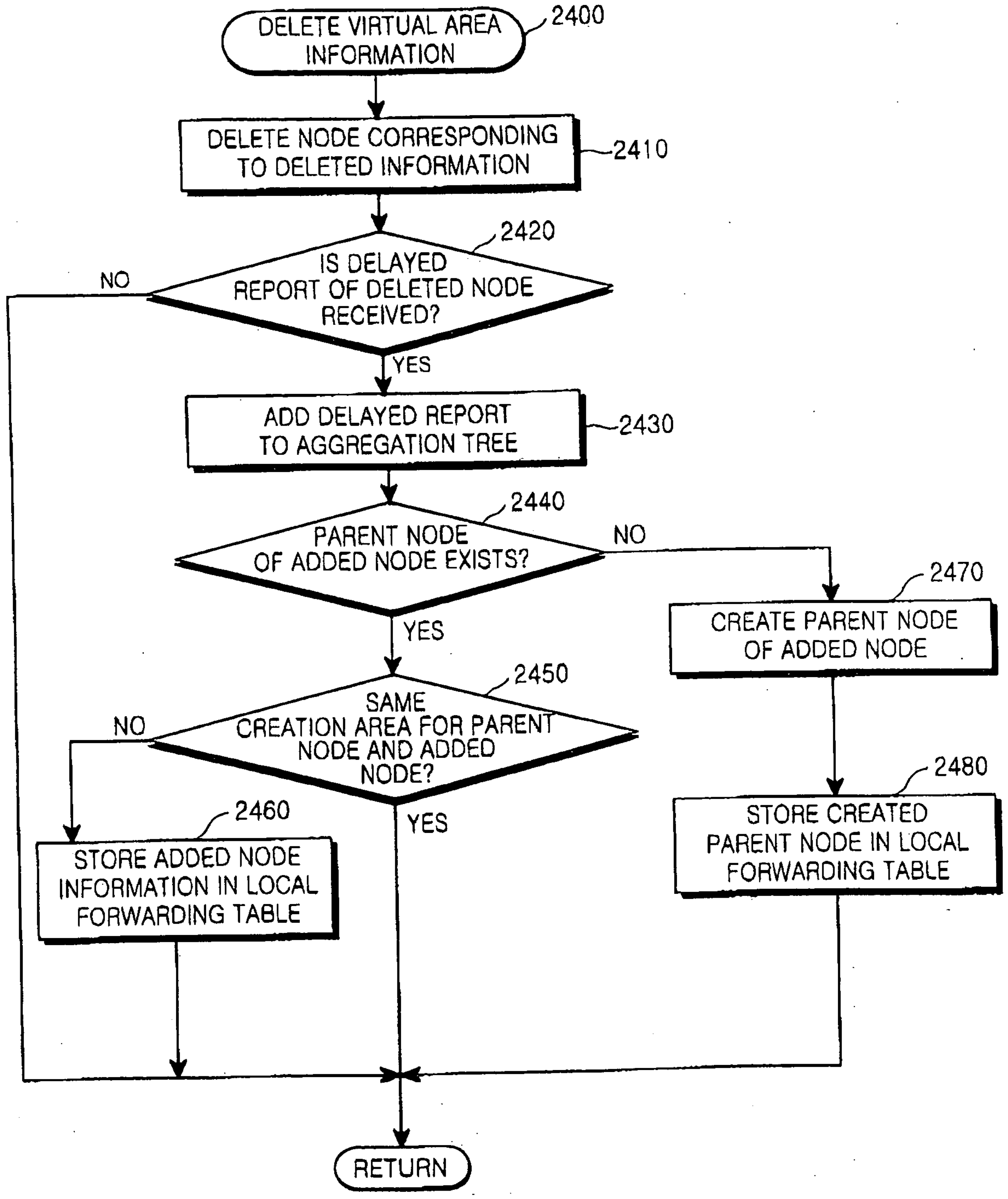
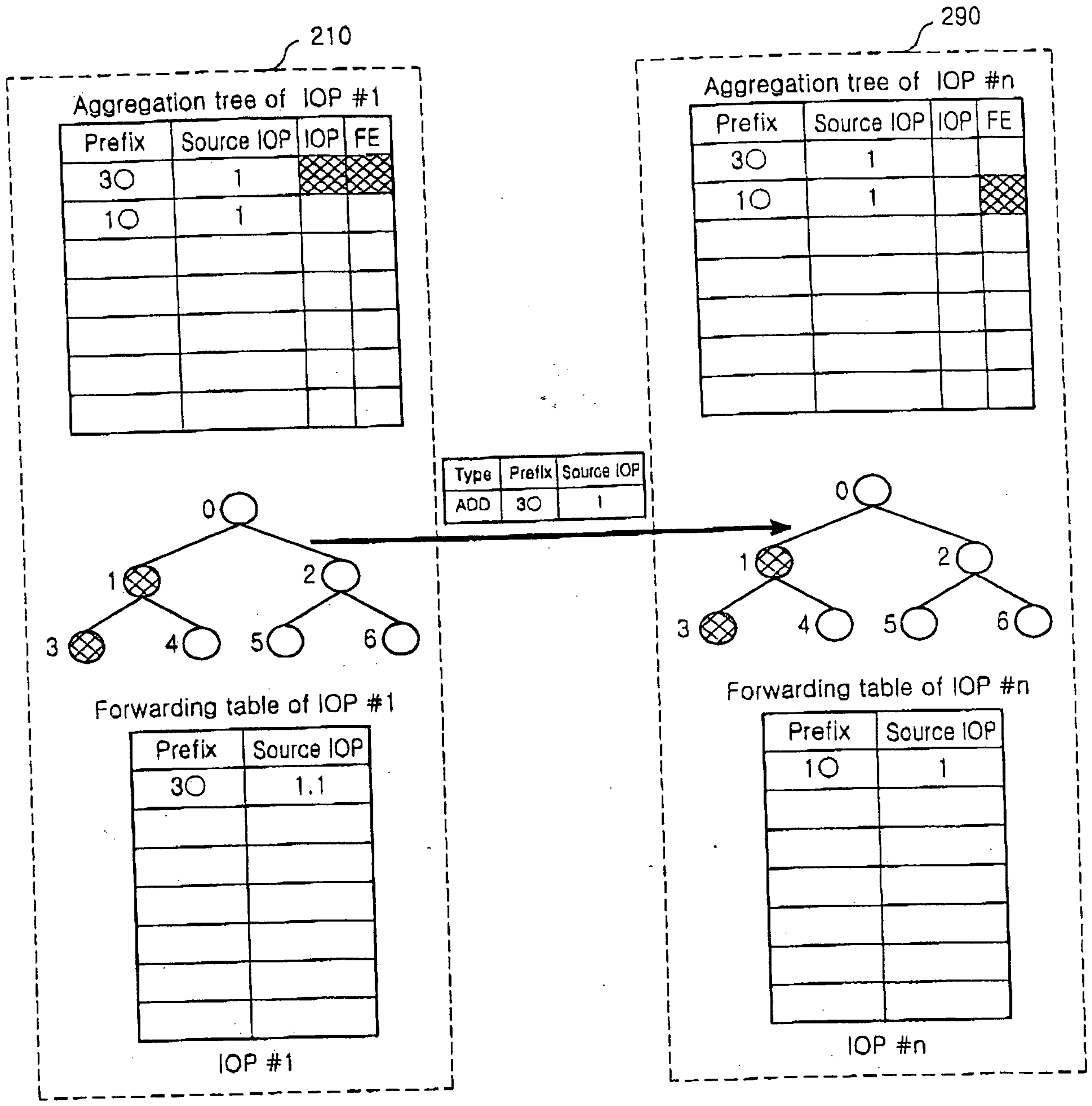

DELETE VIRTUAL AREA INFORMATION
2400
DELETE NODE CORRESPONDING TO DELETED INFORMATION
2410
2420
IS DELAYED REPORT OF DELETED NODE RECEIVED?
NO
YES
ADD DELAYED REPORT TO AGGREGATION TREE
2430
2440
PARENT NODE OF ADDED NODE EXISTS?
NO
YES
2470
CREATE PARENT NODE OF ADDED NODE
2450
SAME CREATION AREA FOR PARENT NODE AND ADDED NODE?
NO
YES
2480
STORE CREATED PARENT NODE IN LOCAL FORWARDING TABLE
2460
STORE ADDED NODE INFORMATION IN LOCAL FORWARDING TABLE
RETURN

FIG.10


210
290
Aggregation tree of IOP #1
Aggregation tree of IOP #n
0
1
2
3
4
5
6
Forwarding table of IOP #1
Forwarding table of IOP #n
IOP #1
IOP #n

Aggregation tree of IOP #1

Prefix	Source IOP	IOP	FE
3○	1	▨	▨
1○	1		

Aggregation tree of IOP #n

Prefix	Source IOP	IOP	FE
3○	1		
1○	1		▨

Type	Prefix	Source IOP
ADD	3○	1

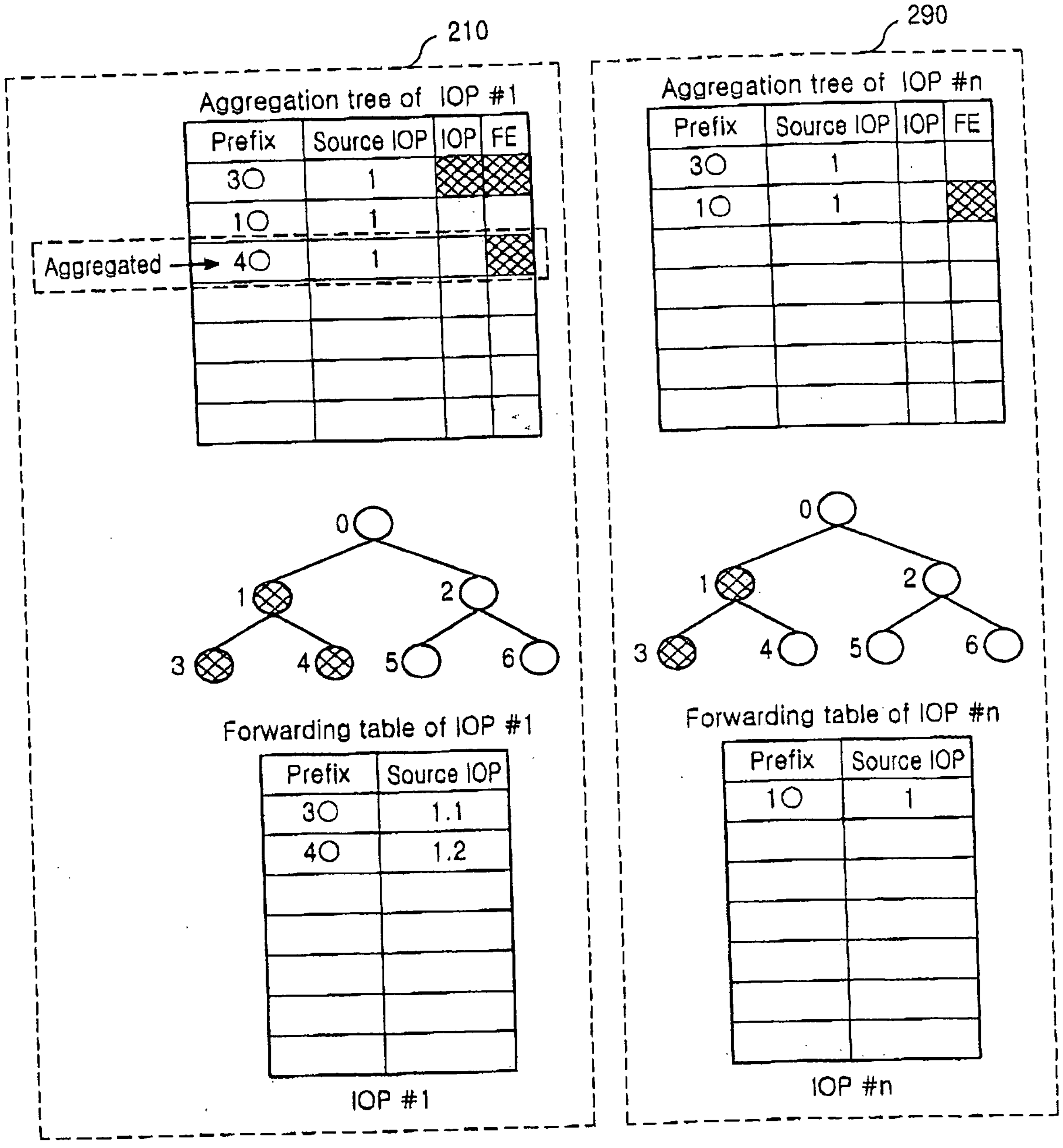
Forwarding table of IOP #1

Prefix	Source IOP
3○	1.1

Forwarding table of IOP #n

Prefix	Source IOP
1○	1

FIG.11A



Aggregation tree of IOP #1

Prefix	Source IOP	IOP	FE
3○	1	▨	▨
1○	1		
4○	1		▨

Forwarding table of IOP #1

Prefix	Source IOP
3○	1.1
4○	1.2

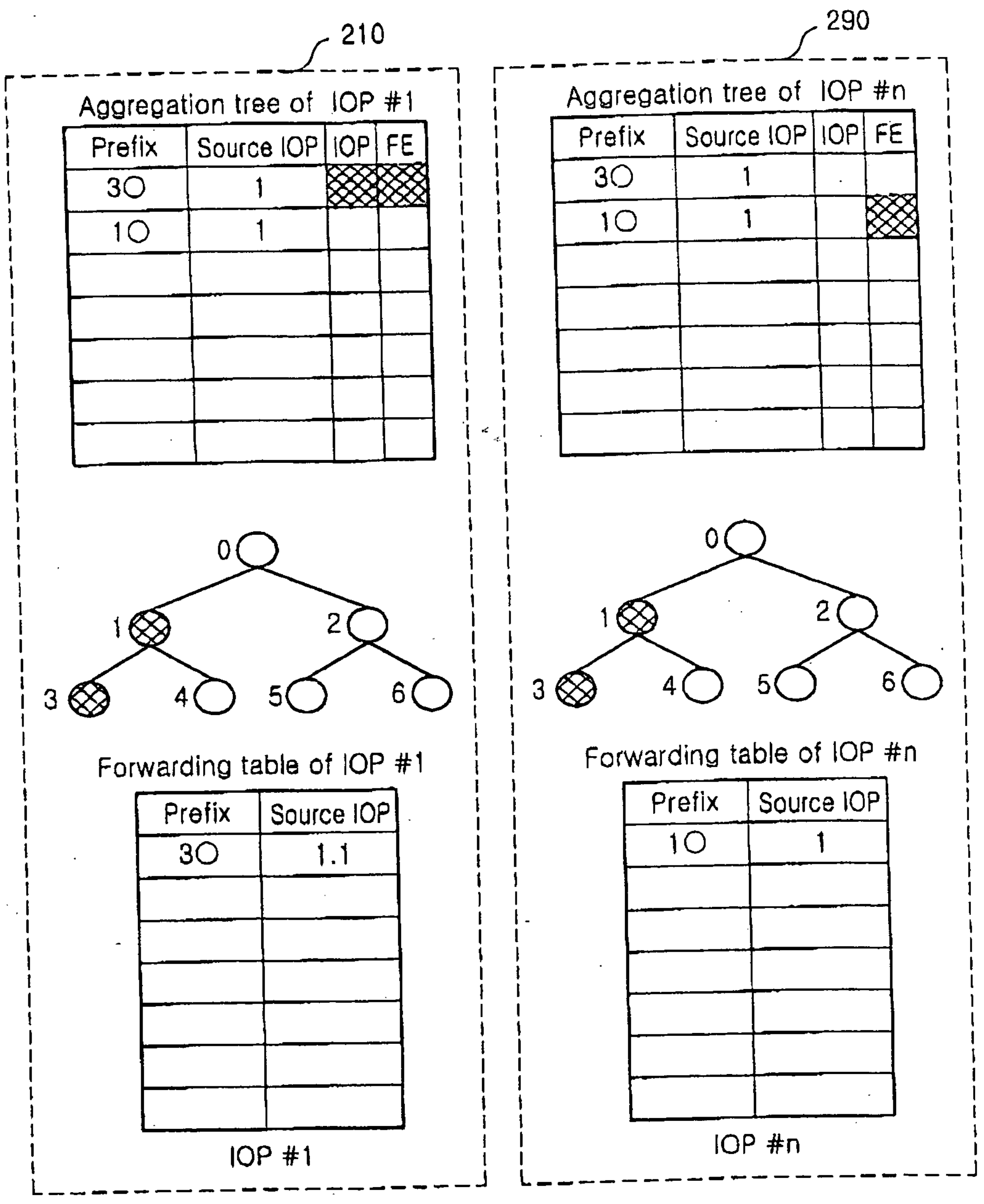
Aggregation tree of IOP #n

Prefix	Source IOP	IOP	FE
3○	1		
1○	1		▨

Forwarding table of IOP #n

Prefix	Source IOP
1○	1

FIG.11B



Aggregation tree of IOP #1

Prefix	Source IOP	IOP	FE
3○	1	▨	▨
1○	1		

Forwarding table of IOP #1

Prefix	Source IOP
3○	1.1

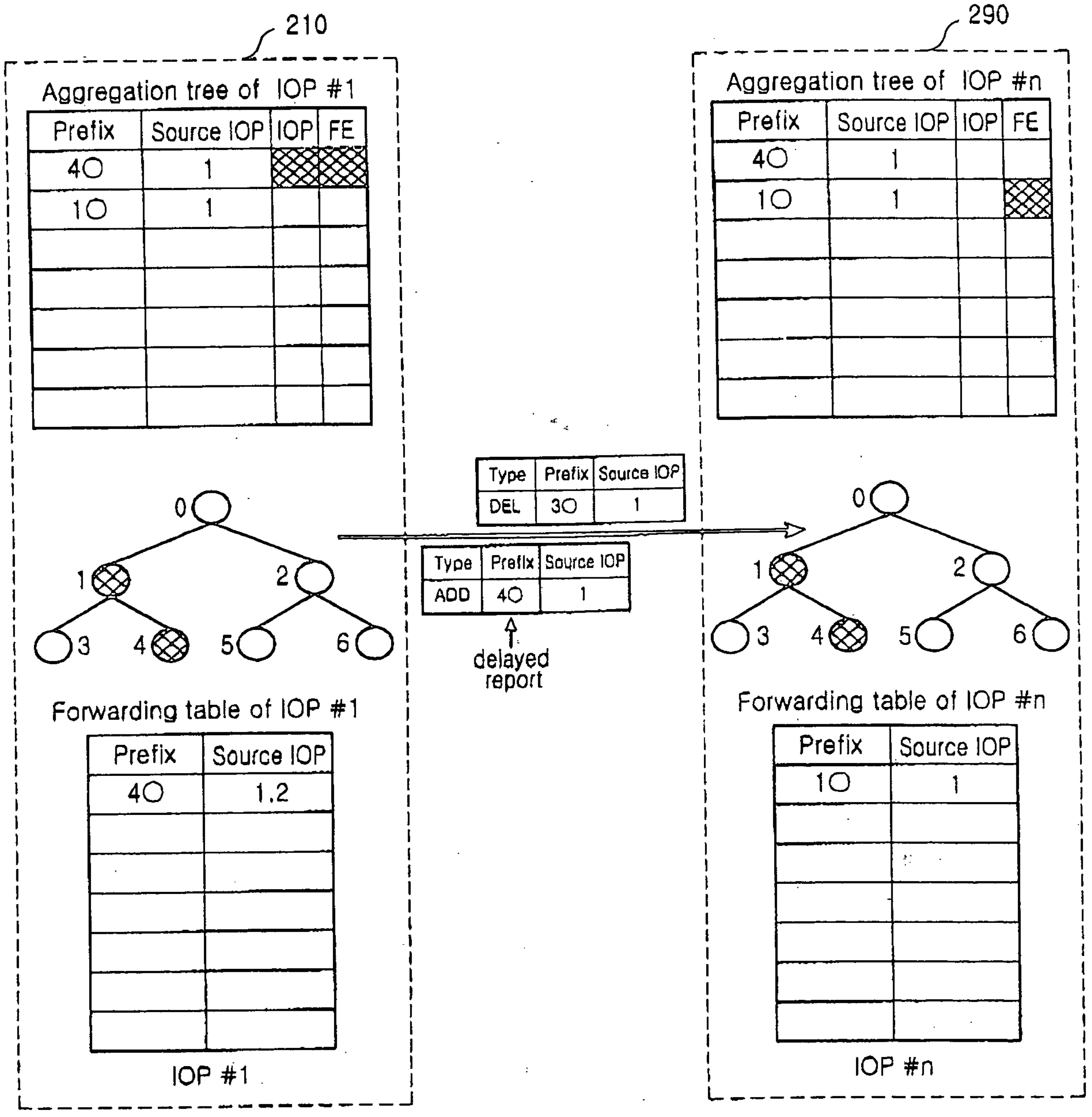
Aggregation tree of IOP #n

Prefix	Source IOP	IOP	FE
3○	1		
1○	1		▨

Forwarding table of IOP #n

Prefix	Source IOP
1○	1

FIG.12A


210
Aggregation tree of IOP #1
0
1
2
3
4
5
6
Forwarding table of IOP #1
IOP #1
delayed report
290
Aggregation tree of IOP #n
0
1
2
3
4
5
6
Forwarding table of IOP #n
IOP #n

Aggregation tree of IOP #1

Prefix	Source IOP	IOP	FE
4○	1	▨	▨
1○	1		

Forwarding table of IOP #1

Prefix	Source IOP
4○	1,2

Type	Prefix	Source IOP
DEL	3○	1

Type	Prefix	Source IOP
ADD	4○	1

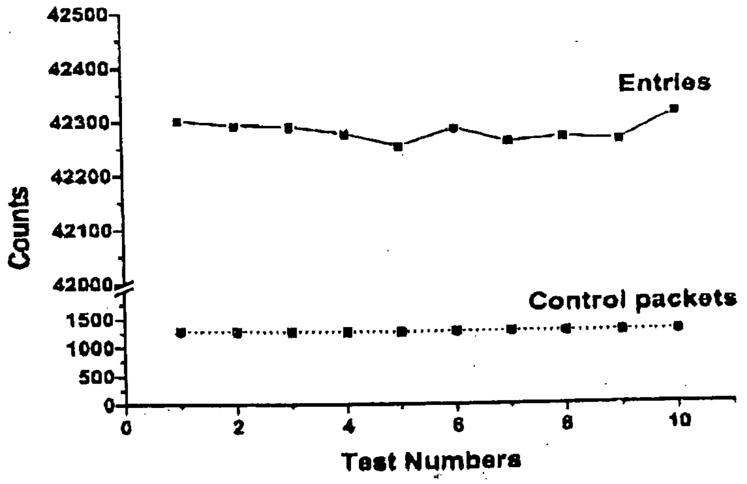
Aggregation tree of IOP #n

Prefix	Source IOP	IOP	FE
4○	1		
1○	1		▨

Forwarding table of IOP #n

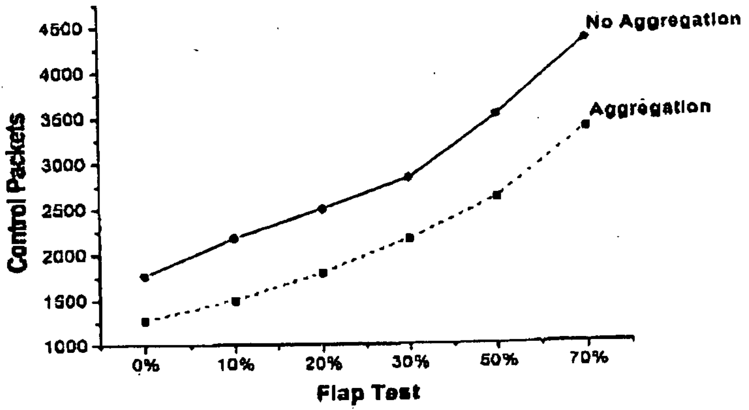
Prefix	Source IOP
1○	1

FIG.12B



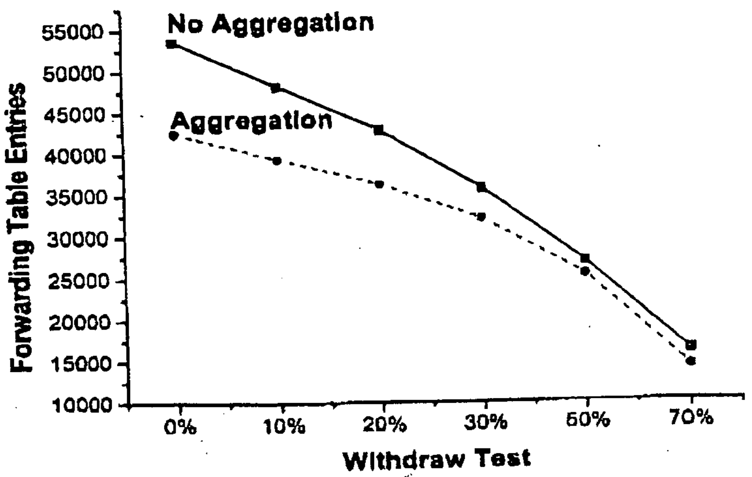
RANDOM ORDER ENTRY TEST

FIG.13A



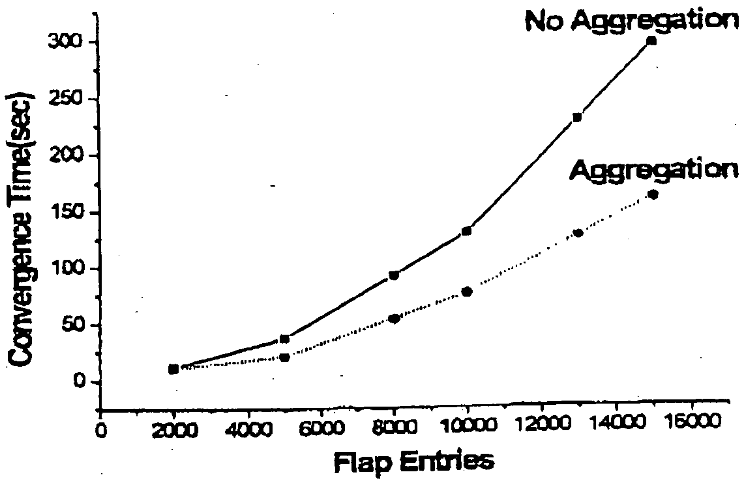
CONTROL PACKET TRANSMISSION

FIG.13B



NUMBER OF ENTRIES IN FORWARDING TABLE

FIG.13C



BGP4 FLAP CONVERGENCE TIME TEST

FIG.13D

```
Insertion (prefix) {
 local insertionnode
 /* STEP1 */
 insertionnode := FindNode(prefix)
 /* STEP2 */
 /* Check new route's nexthop */
 if (NodeNexthopVirtual(insertionnode) = TRUE) then
    InsertionNodeVirtual(insertionnode)
 else
    InsertionNodeInter-domain(insertionnode)
 /* STEP3 */
 if (run_step3 = TRUE) then
    ChildNodeHandler(insertionnode)
}
```

FIG.14A

```
Deletion (prefix) {
  local deletenode, siblingnode
  /* STEP1 */
  deletenode := FindNode(prefix)
  /* STEP2 */
  if (DRNForward(deletenode) = TRUE) then
    if (ForwardingTableForward(deletenode) = TRUE)
      then
        DeleteForwardingTable(deletenode)
        SendDRN(deletenode)
      else
        /* Deletion of the parent node instead of  */
        /* the deletenode                          */
        DeleteForwardingTable(GetParent(deletenode))
  else
     /* In case of the deletion of the aggregation node, */
     /* sending information to DRN can be suppressed */
     if (ForwardingTableForward(deletenode) = TRUE)
      then
        DeleteForwardingTable(deletenode)

  siblingnode := SiblingNodeCheck(deletenode)
  / * Delayed insertion of the sibling node */
  if (siblingnode ≠ NIL) then
      SendDRN(siblingnode)
  /* STEP3 */
  ChildNodeHandler (deletenode)
}
```

FIG.14B

```
FindNode (prefix) {
  local node
  /* Search for node to be inserted */
  node := GetNode (prefix)
  /* Identity the insertion node */
  if ((node ? NIL) and (NodeType(node) = AGG)) then
        Empty (node)
        run_step3 := TRUE
  else
        NewNode(node)
  return (node)
}
```

FIG.14C

```
MakeParentNode (node) {
  local parentnode
  /* Make parent node and set node type to AGG */
  parentnode := AllocateParentNode(node)
  NodeType(parentnode) := AGG
  Parent(node) := parentnode
  return(parentnode)
}
```

FIG.14D

```
InsertNodeVirtul (node) {
 local parentnode
 parentnode := GetParent(node)
 if (parentnode ? NIL) then
  if (NodeSource(parentnode) ? NodeSource(node)) then
        InsertForwardingTable(node)
  /* There is not parent node. */
 else
  /* Make and wirte parent node to the forwarding table */
     parentnode := MakeParentNode(node)
     InsertForwardingTable(parentnode)
}
```

FIG.14E

```
InsertNodeInter-domain (node) {
 local parentnode
 parentnode := GetParent(node)
 if (parentnode ? NIL) then
     /* If new route source and parent are same, */
     /* new route sending to DRN can be suppressed */
  if (NodeSource(parentnode) ? NodeSource(node)) then
        SendDRN(node)
     InsertForwardingTable(node)
 else
     /* Make parent node and wirte new node */
/* to the forwarding table */
     parentnode := MakeParentNode(node)
     InsertForwardingTable(node)
     SendDRN(node)
}
```

FIG.14F

```
ForwardingTableForwardCheck(node)
{
  local parentnode
  parentnode := GetParent(node)
  if ((ForwardingTableForward(node) = FALSE)
  and (NodeSource(node) != NodeSource(parentnode))) then
        return (FALSE)
  else
      return (TRUE)
}
```

FIG.14G

```
DRNForwardCheck(node)
{
  local parentnode
  parentnode := GetParent(node)
  if ((DRNForward(node) = FALSE)
     and (NodeNexthopVirtual(node) = FALSE)) then
      return (FALSE)
  else
      return (TRUE)
}
```

FIG.14H

```
ChildNodeHandler (node) {
 local leftchild, rightchild
 leftchild := GetLeftChildNode(node)
 rightchild := GetRightChildNode(node)
 if (leftchild ? NIL) then
     /* Disaggregation of the leftchild node */
   if ((ForwardingTableForwardCheck(leftchild) = FALSE)  then
           InsertForwardingTable(leftchild)
      /* Delayed insertion of the leftchild node */
      if ((DRNForwardCheck(leftchild) = FALSE) then
           SendDRN(leftchild)
   if (rightchild ? NIL) then
      /* Disaggregation of the rightchild node */
   if ((ForwardingTableForwardCheck(rightchild) = FALSE) then
           InsertForwardingTable(rightchild)
       /* Delayed insertion of the rightchild node */
      if ((DRNForwardCheck(rightchild) = FALSE) then
           SendDRN(rightchild)
}
```

FIG.14I

```
SiblingNodeCheck (node) {
  local siblingnode
  siblingnode := GetSiblingNode(node)
  /* Check if there is aggregated sibling node */
  if ((NodeSource(node) = NodeSource(siblingnode))
      and (DRNForward(siblingnode) = FALSE)) then
       return (siblingnode)
  else
       return (FALSE)
}
```

FIG.14J